IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	Duane A. Averill, et al.	:	Date: June 4, 2007
Group Art Unit:	2188	:	IBM Corporation
Examiner:	M. Chery	:	Intellectual Property Law
Serial No.:	10/760,431	:	Dept. 917, Bldg. 006-1
Filed:	January 20, 2004	:	3605 Highway 52 North
Title:	METHOD AND APPARATUS FOR TRACKING CACHED ADDRESSES FOR MAINTAINING CACHE COHERENCY IN A COMPUTER SYSTEM HAVING MULTIPLE CACHES	:	Rochester, MN 55901

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 223313-1450

APPEAL BRIEF IN SUPPORT OF APPEAL
FROM THE PRIMARY EXAMINER TO THE BOARD OF APPEALS

Sir:

This is an appeal of a Final Rejection under 35 U.S.C. §103(a) of claims 1-22 of Application Serial No. 10/760,431, filed January 20, 2004. This brief is submitted pursuant to a Notice of Appeal filed April 2, 2007, as required by 37 C.F.R. §1.192.

1. Real Party in Interest

International Business Machines Corporation of Armonk, NY, is the real party in interest. The inventors assigned their interest as recorded on January 20, 2004 on Reel 014919, Frame 0245.

2. Related Appeals and Interferences

There are no related appeals nor interferences pending with this application.

3. Status of Claims

Claims 1-22 are pending and stand finally rejected. The claims on appeal are set forth in the Appendix of Claims

4. Status of Amendments

A response was submitted following final rejection, but it contained only argument and did not amend any claim. By Advisory Action dated March 9, 2007, the Examiner indicated that the response would be entered but would not place the application in condition for allowance.

5. Summary of Claimed Subject Matter

The invention herein relates to maintaining cache coherency among multiple caches, and is particularly intended for use where one of the caches has a relatively high turnover. Independent claims 1, 9 and 19 recite respectively a digital data processing system, method and cache coherency apparatus, wherein at least part of a cache line state directory structure has a fixed number of entries each having a fixed correspondence to a unique respective one of the slots in a corresponding cache. Independent claim 14 recites a digital data processing system having essentially similar limitations, and further that

another part of the directory structure contains entries having a fixed correspondence to real addresses.

.

In accordance with claim 1, a digital data processing system includes a memory, at least one processor having a cache, and a device having a device cache. (Spec. p. 6, lines 19-23; p. 9, line 13 - p. 13, line 6; Figs 1 & 2, features 102, 103, 205, 206 207, 210). The device cache has a fixed number of slots for caching data (Spec. p. 6, lines 20-23). A cache coherency mechanism includes a cache line state directory structure, wherein at least part of the cache line state directory structure associated with the device contains a fixed number of entries equal to the fixed number of slots in the device cache, there being a one-to-one correspondence between these directory structure entries and the slots of the device cache (Spec p. 6, lines 20-23; p. 13, lines 25-27; p. 18, line 21 - p. 19, line 10; Fig. 3, feature 305, Fig. 5). The cache coherency mechanism selectively determines whether to send cache line invalidation messages to the device using the cache line state directory structure (Spec. p. 6, lines 23-24; p. 15, lines 18-19; p. 21, lines 14-24 Fig. 7, step 705; Fig. 9, step 904)

In accordance with claim 9, at least a portion of a cache line state directory structure, wherein at least part of the cache line state directory structure corresponding to a device cache contains exactly N entries equal to exactly N slots in the device cache, there being a one-to-one correspondence between these directory structure entries and the slots of the device cache (Spec p. 6, lines 20-23; p. 13, lines 25-27; p. 18, line 21 - p. 19, line 10; Fig. 3, feature 305, Fig. 5). Responsive to data access requests, the cache line state directory structure is accessed to determine whether the data is in the device cache (Spec. p. 6, lines 23-24, p. 15, lines 18-19; p. 19, lines 11-21; Fig. 5). A determination is

made whether to send a cache line invalidation message based on the result (Spec. p. 6, lines 23-24; p. 15, lines 18-19; p. 21, lines 14-24 Fig. 7, step 705; Fig. 9, step 904).

In accordance with claim 14, a digital data processing system includes a memory, multiple processors controlling multiple caches, and a device having a device cache. (Spec. p. 6, lines 19-23; p. 9, line 13 - p. 13, line 6; Figs 1 & 2, features 102, 103, 205, 206 207, 210). The device cache has a fixed number of slots for caching data (Spec. p. 6, lines 20-23). A cache coherency mechanism includes a cache line state directory structure, wherein a first part of the directory structure contains multiple entries, each having a fixed correspondence to a respective set of real addresses, and a second part of the cache line state directory structure associated with the device contains a fixed number of entries equal to the fixed number of slots in the device cache, there being a one-to-one correspondence between these directory structure entries and the slots of the device cache (Spec p. 6, lines 20-23; p. 13, lines 25-27; p. 16, line 21 - p. 17, line 10; p. 18, line 21 - p. 19, line 10; Fig. 3, feature 305, Figs 4 and 5).

In accordance with claim 19, a cache coherency apparatus for a digital data processing system includes a communications interface and a cache line state directory structure, wherein at least part of the cache line state directory structure associated with the device contains a fixed number of entries equal to the fixed number of slots in the device cache, there being a one-to-one correspondence between these directory structure entries and the slots of the device cache (Spec p. 6, lines 20-23; p. 13, lines 21-27; p. 18, line 21 - p. 19, line 10; Fig. 3, features 301-305, Fig. 5). The cache coherency mechanism selectively determines whether to send cache line invalidation messages to the device using the cache line state directory structure (Spec. p. 6, lines 23-24; p. 15, lines 18-19; p. 21, lines 14-24 Fig. 7, step 705; Fig. 9, step 904)

6. Grounds of Rejection To Be Reviewed on Appeal

Claims 1, 2 and 5-8 are finally rejected under 35 U.S.C. §103(a) as unpatentable over Baumgartner (US 6,108,764) in view of Arimilli (US 2003/0009643). Claims 3-4 and 9-22 are finally rejected under 35 U.S.C. §103(a) as unpatentable over *Baumgartner* and *Arimilli* in view of Carpenter et al. (US 6,115,804). The only issues in this appeal are whether the claims are prima facie obvious over the cited references.

7. Argument

Appellants contend that the Examiner failed to establish adequate grounds of rejection for the following reasons:

I. The Examiner improperly rejected all claims under 35 U.S.C. §103(a) because neither *Baumgartner* nor *Arimilli* (nor the tertiary reference), considered alone or in combination, discloses the key features of appellant's independent claims, i.e., a cache line state directory used to send invalidation messages to a device, having a fixed one-to-one correspondence with slots in the device cache. [page 9 below]

II. The Examiner additionally improperly rejected independent 12 (and claims dependent on it, as well as dependent claims 3, 4 and 21) under 35 U.S.C. §103(a) because neither none of the references, considered alone or in combination, discloses a two-part cache line state directory, one part having correspondence to cache slots and the other to real addresses. [page 15 below]

Overview of Invention

A brief overview of appellants' invention in light of existing art will be helpful in appreciating the issues herein. Appellants' invention relates to cache coherency in a computer system having multiple caches, and is of particular application to a computer

system utilizing a so-called Non-Uniform Memory Access (NUMA) architecture.[1] In a typical NUMA architecture, main memory is distributed, a respective discrete subset of main memory being associated with each node, processor or group of processors. Each processor, group of processors or node may additionally cache portions of main memory in one or more hierarchical levels. Where multiple caches exist in different nodes, cache coherency becomes a significant architectural issue.

There are various techniques in the art for supporting cache coherency where multiple caches exist. In many smaller systems, an invalidation message is broadcast whenever data is stored or altered, and each cache receiving such a message determines by examining its directory whether the message is of interest. Broadcasting is relatively simple, but is not well adapted to be scaled for very large systems. As the number of caches grows, the number of invalidation messages broadcast grows proportionately, and these tend to clog the communications channels. In many cases, no other cache has a copy of the data, so the message is entirely unnecessary. But the broadcasting entity has no way of knowing that.

For a large system, particularly a NUMA system, there is a need to reduce the volume of invalidation messages. A known technique is to make each node of the NUMA system responsible for tracking those memory addresses in its subset of main memory which are (or might be) in a cache, either in the node which tracks it or in some other node. This information is tracked in a structure which appellants call a "cache line state directory". If an action is taken which might affect validity of cache data (e.g., a store to main memory), the node containing the memory address checks its cache line

[1] The independent claims herein are not limited to a NUMA architecture, but the background discussion is useful in understanding the motivation for appellants' invention.

state directory to determine whether a copy may exist elsewhere in cache, and transmits an invalidation message only if such a copy may exist. It is important to understand that a function of such a cache line state directory is therefore to reduce unnecessary invalidation messages.

The data in the cache line state directory is not necessarily perfectly current, but is sufficient for its purpose. In particular, when a line of data is loaded into a cache, a corresponding reference is placed in the cache line state directory, so that it will always be possible to find a reference to a line which is in the cache. But the reverse is not necessarily true immediately if a line is overwritten or invalidated in a cache. In the latter case, the reference may linger in the cache line state directory, although due to any of various mechanisms it is eventually overwritten with new data. Conventionally, the cache line state directory is set associative, accommodating multiple entries for each hashed or otherwise encoded portion of an address used to access cache. I.e., ***the entries in a conventional cache line state directory correspond to real address ranges*** (this makes them easier to locate, given an input real address). The set associativity must be sufficiently large so that, when a line is loaded to cache, there will be an available entry in the associativity set corresponding to the cache line address for a reference to the loaded line. It is also possible to design variable sized associativity set directory structures.

Appellants observed that certain device caches, such as a cache for an I/O bridge device, tend to have a very high turnover in a relatively small cache which acts as a buffer, and as a result of the conventional set associative design, can rapidly displace a large number of references in a cache line state directory. This causes various inefficiencies in the use of cache line state directory space, which in turn causes any of

various secondary mechanisms to prematurely invalidate cache lines or take other actions which are undesirable from a performance standpoint.

Since the cache in an I/O bridge device or certain similarly behaving devices is often quite small, appellants addressed this problem by providing a separate cache line state directory portion, in which the entries do not correspond to real addresses (as in the case of a conventional cache line state directory), but instead have a fixed one-to-one correspondence to the slots in the device cache. Thus, when a new cache line is loaded to a slot in the device cache, the corresponding entry in the cache line state directory is updated, automatically overwriting any previous obsolete entry. As a result, the number of references required to be kept for the device cache can never exceed the number of slots in the device cache (unlike the case of the conventional set associative cache). This approach requires additional hardware to simultaneously search all the cache line state directory entries which correspond to slots in the device cache, but it is practical if the number of such slots is relatively small. It effectively prevents the small, high-turnover device cache from displacing a much larger number of entries in the cache line state directory.

Therefore a significant feature of all independent claims herein is that a cache line state directory ***which is used to determine whether to send invalidation messages*** to a device contains ***entries having a fixed one-to-one correspondence with the slots*** in a device cache.

I. The Examiner improperly rejected all claims under 35 U.S.C. §103(a) because neither *Baumgartner* nor *Arimilli* (nor the tertiary reference), considered alone or in combination, discloses the key features of appellant's independent claims, i.e., a cache line state directory used to send invalidation messages to a device, having a fixed one-to-one correspondence with slots in the device cache.

In order to support a rejection for obviousness, there must be some suggestion in the art to combine the references in such a manner as to form each and every element of appellants' claimed invention. It is not sufficient that a suggestion may exist to combine the references, if such a combination does not meet the limitations of appellants' claims without some further non-obvious modification. Assuming arguendo that a suitable suggestion exists in the art to combine the references, the hypothetical combination fails to teach or suggest the significant claimed features of appellants' invention, and therefore the rejection was improper.

Appellants' representative claim 1 recites:

1. A digital data processing system, comprising:
a memory;
at least one processor having at least one associated cache for temporarily caching data from said memory;
at least one device having a device cache, said device cache having a fixed number of slots for caching data, said fixed number being greater than one, each slot caching a cache line of data; and
a cache coherency mechanism, said cache coherency mechanism including a cache line state directory structure, said cache coherency mechanism *selectively determining whether to send cache line invalidation messages to said at least one device using state information in said cache line state directory structure*, wherein at least a portion of said cache line state directory structure *associated with said at least one device contains exactly said fixed number of cache line entries, each entry having a fixed correspondence to a unique respective one of said fixed number of slots* for caching data of said device cache. [emphasis added]

The remaining independent claims, while not identical in scope, all contain limitations analogous to the italicized language above.

As a preliminary matter, appellants have in responding to the rejections characterized the above italicized limitation as a "one-to-one correspondence" between slots in the device cache and entries in the cache line state directory. The Examiner has taken the position that appellants' claims do not recite a one-to-one correspondence. It is true that appellants' claims do not use the precise words "one-to-one correspondence" but appellants submit that their characterization of the claim limitation is accurate. The exact words of claim 1 are: "...contains exactly said fixed number of cache line entries [i.e. the number of slots and cache line entries is fixed and identical], each entry having a ***fixed correspondence to a unique respective one*** of said fixed number of slots for caching data of said device cache." This is effectively a limitation of a one-to-one correspondence; how else would one define it? Appellants have used the term "one-to-one correspondence" because it is easier to understand and apply that a lengthy repetition of the exact claim language in each instance. But even if one does not accept the characterization as a "one-to-one correspondence", the fact is that this limitation, however called, is not disclosed or suggested by the cited art, for the reasons explained below.

Baumgartner discloses a NUMA computer system having multiple caches. The thrust of *Baumgartner* appears to be a mechanism whereby in some circumstances a cache for one processor can satisfy a read request from a different processor, thereby reducing latency time for memory reads. *Baumgartner* further discloses a cache coherency mechanism including a "coherence directory", which stores indications of memory address of data checked out to caches in remote nodes. The detailed structure of the "coherence directory" is not disclosed, although it appears to be organized as a conventional set-associative directory in which entries correspond to memory addresses. Whatever its organization, there is no disclosure or suggestion whatsoever in *Baumgartner* that the "coherence directory" be organized as a directory having multiple entries, each entry having a fixed correspondence to a respective unique slot of a device

cache to which it corresponds.[2] The Examiner apparently concedes as much in his rejection, for he relies on *Arimilli* for a teaching of this latter aspect.

Arimilli discloses a NUMA system architecture having multiple processors in each node, each processor having its own portion of memory and one or more caches. *Arimilli*'s invention appears to be directed to the reduction of queue length in a node controller, by delaying placing certain memory access operations on the queue. Specifically, a memory access generated by a processor within a node is not placed on the queue for remote operations until all local processors have responded and the requested data has not been found in the memory portions or caches associated with the local processors. *Arimilli* further discloses a cache coherence mechanism, including a respective "local memory directory" associated with each processor. The "local memory directory" appears to be a form of cache line state directory, i.e., it is used to maintain cache line state information for data in the memory portion associated with the processor, and which might be held in a cache of another processor in the same or a different node. However, *Arimilli*'s local memory directory appears to be set-associative; at least, there is no disclosure of a fixed one-to-one correspondence between the entries of the local memory directory and entries in the caches of other processors or devices.

---

[2] See, e.g., the following passages from *Baumgartner*:

Coherence directory 50 stores indications of the system memory addresses of data (e.g. cache lines) checked out to caches in remote nodes for which the local processing node is the home node. The address indication for each cache line is stored in association with an *identifier of each remote processing node* having a copy of the cache line and the coherency status of the cache line at each such remote processing node... [Col. 7, lines 58-66, emphasis added]

The use of the phrase "each remote processing node" indicates that a single entry in the coherence directory corresponds to data (rather than cache slots), because if there were a one-to-one correspondence with cache slots as claimed by appellants, then a single entry could not correspond to data in multiple remote nodes.

In the rejection, the Examiner cites a passage from *Arimilli* which discloses a cache directory structure, ***which is separate from, and has a different function than, Arimilli's local memory directory***. The cache directory is a data structure which identifies the address and state of data currently held in the cache. It is used when a processor issues a read request to the cache to determine whether the requested data is actually in the cache, and to identify the corresponding slot of the requested data if it is in fact in the cache. The state data in the cache directory can also be used for certain maintenance operations, e.g., for determining which line to evict when it is necessary to bring a new line of data into the cache. It is therefore crucial to understand that a cache directory structure as disclosed in *Arimilli* is not a cache coherency mechanism. *Arimilli*'s cache directory is generally conventional in design, being set-associative and accessed by hashing an address. There is a one-to-one correspondence between entries of the cache directory and cache lines of the cache, as in conventional design. This is the way virtually all large caches are organized. Indeed, it could hardly be otherwise, for the purpose of the directory is to identify what is in the lines of the cache.

The Examiner reasons that one-to-one correspondence of a directory and a cache structure is thus shown, and therefore it would have been obvious to combine this design feature with the cache coherency structure disclosed in *Baumgartner*. Appellants respectfully disagree.

Baumgartner's "coherence directory" provides a function similar to that provided by appellants' "cache line state directory", but *Arimilli*'s "cache directory" manifestly does not. ***It is an entirely unrelated structure, which does not provide any sort of function similar to that recited in appellants' claims***, i.e., that of determining whether to send an invalidation message to the associated cache. It is used, as noted above, for determining whether data sought by a read request is actually in the associated cache.

The design of a system element is dictated by its function. Conventionally a cache directory is used to look up not only whether a particular cache line exists in a particular cache, but to find the exact cache slot. It has a one-to-one correspondence to the cache slots for this very reason, because that is the most practical way to design it. Conventionally, a coherence structure is designed differently because it has a different function. It is used to determine, for a given real address, whether it might be necessary to broadcast an invalidation message. A one-to-one correspondence with all the possible caches would mean an extremely large and cumbersome directory structure, containing much more information than is necessary for the limited purposes of the coherence structure. For example, for this purpose it is simply unnecessary to know the slot in which data is stored in a particular cache.

Appellants do not recite an arbitrary structure, but one which is used for a specific purpose, i.e., ***"selectively determining whether to send cache line invalidation messages to said at least one device"***. *Arimilli*'s cache directory manifestly is not used for that purpose.

The Examiner is arbitrary mixing design parameters of one functional element with design parameters of another. Appellants concede that a logic designer of ordinary skill in the art could have designed a cache line state directory having a one-to-one correspondence of selective entries with cache lines of a device cache, ***given the proper motivation and direction to do so.*** But the point is that the ordinary function of the cache line state directory does not provide such a motivation, and all prior art structures having similar function are organized differently. The Examiner points to a structure, having a wholly different function, which does indeed have a one-to-one correspondence to cache slots. But the existence of such a structure doe not disclose the motivation to modify a conventional cache coherence structure as claimed by appellants.

Ultimately, one must ask wherefrom comes the motivation to construct a coherence directory having one-to-one correspondence with one of the device caches. As explained previously, the motivation for appellants' invention is to avoid a situation in which a cache in an I/O bridge or similar device, having a small size and very large turnover, crowds out the other entries in the coherence directory structure. The entries for the I/O bridge are typically very short lived, so that at any given instant the vast majority of such entries would be already invalidated. If a cache line state directory is thus filled with I/O bridge entries pointing to invalidated lines, it becomes largely useless as a performance enhancement mechanism. This motivation is neither disclosed nor suggested by any of the cited references. It comes from one place and one place only: appellants' disclosure. By thus modifying a *Baumgartner*'s coherence directory to structure it having a one-to-one correspondence with cache slots, the Examiner is relying on hindsight gleaned from appellants' disclosure.

The tertiary reference, *Carpenter*, similarly fails to teach or suggest the key limitations of appellants' claims as discussed above. *Carpenter* discloses a NUMA system similar in design to that discloses in *Baumgartner*, further disclosing a "coherence directory" which, like *Baumgartner*'s, appears to be organized along conventional lines. There is nothing in *Carpenter* to suggest organizing some portion of the coherence directory as a one-to-one correspondence of directory entries to cache slots.

For all the reasons explained above, the combination of the cited references fails to teach or suggest the critical limitations of appellants' claims, and the rejections of the claims were erroneous.

II. The Examiner additionally improperly rejected independent 12 (and claims dependent on it, as well as dependent claims 3, 4 and 21) under 35 U.S.C. §103(a) because neither none of the references, considered alone or in combination, discloses a two-part cache line state directory, one part having correspondence to cache slots and the other to real addresses.

In addition to the limitations discussed in part I above with respect to all claims, independent claim 12, as well as dependent claims 3, 4 and 21, contain the limitation that the cache line directory structure contains multiple portions, one of which is organized having the one-to-one correspondence with cache slots discussed above, and the other of which has entries corresponding to real addresses. I.e., the second portion is organized in the conventional manner for a coherence structure. The division of the cache coherence structure into multiple parts having different organization is not disclosed or suggested by any of the cited art.

A design which splits a cache coherence directory into multiple portions, each differently organized, it not in itself difficult to implement, but again there must be a motivation to do so. All of the cited art shows various directory structures having a single organization. Single organization is conventional because each structure normally has a particular identified function and, given the system architecture, a single optimal organization is chosen according to any of various system parameters.

For the reasons explained above, appellants determined that in certain circumstances there is a benefit to dividing the coherency directory into multiple differently organized structures, i.e., the entries of a single uniform structure become filled with the high-turnover cache of a bridge I/O or similar device. Splitting up the cache coherency directory avoids this.

Such a division of the cache coherency structure runs counter to the normally assumed optimal design. Typically, the entire coherency structure is organized with entries corresponding to memory addresses. In this way, if a particular line is checked out to multiple caches, only a single entry is required. Furthermore, by sharing the entries in a single structure, irregular distribution in one cache tends to be cancelled out by random distribution of cache lines in other caches, resulting in a more efficient use of the available entries.

Appellants' invention amounts to a rejection of conventional wisdom and the more commonly applied approach. Notwithstanding that the ability to design logic circuits implementing such a design is well within the capabilities of those of skill in the art, there is no motivation shown in the references or in the art in general to do so. The motivation comes entirely from appellants' observations concerning the undesirable effects of monopolization of coherency entries in certain circumstances. These observations are not general knowledge in the art, nor are they disclosed in any of the cited references. For these reasons, there would have been no motivation for one of skill in the art to implement a design having multiple coherency directory sections, each organized differently, as recited in claim 12 and the various dependent claims.

For all of these reasons, the Examiner's rejections of claims 3, 4, 12-18 and 21 were improper.

8. Summary

Appellants disclose and claim a novel and unobvious design for a coherency structure called a "cache line state directory", in which at least a portion of the directory contains entries having a fixed, one-to-one correspondence with slots in a device cache.

The prior art discloses the use of coherency structures organized differently, in which entries correspond to memory addresses. Although conventional structures exist which have the property of one-to-one correspondence to cache slots, these structures perform different function. No motivation is shown in the references or in the art to design a cache coherency structure, used for sending invalidation messages, having one-to-one correspondence to cache slots. The motivation for such a design is taught entirely by appellants.

For all the reasons stated herein, the rejections for obviousness were improper, and appellants respectfully requests that the Examiner's rejections of the claims be reversed.

Date: June 4, 2007

Respectfully submitted,

DUANE A AVERILL, et al.

By_____

Roy W. Truelson, Attorney
Registration No. 34,265
(507) 202-8725 (Cell) (507) 289-6256 (Office)

From: IBM Corporation
Intellectual Property Law
Dept. 917, Bldg. 006-1
3605 Highway 52 North
Rochester, MN 55901

APPENDIX OF CLAIMS

1. A digital data processing system, comprising:

a memory;

at least one processor having at least one associated cache for temporarily caching data from said memory;

at least one device having a device cache, said device cache having a fixed number of slots for caching data, said fixed number being greater than one, each slot caching a cache line of data; and

a cache coherency mechanism, said cache coherency mechanism including a cache line state directory structure, said cache coherency mechanism selectively determining whether to send cache line invalidation messages to said at least one device using state information in said cache line state directory structure, wherein at least a portion of said cache line state directory structure associated with said at least one device contains exactly said fixed number of cache line entries, each entry having a fixed correspondence to a unique respective one of said fixed number of slots for caching data of said device cache.

2. The digital data processing system of claim 1, wherein said device is an I/O bridge device.

3. The digital data processing system of claim 1, wherein a processor portion of said cache line state directory structure contains cache line state for at least one said cache associated with a processor, said processor portion being separate from said at least a portion of said cache line state directory structure associated with said at least one device, said processor portion containing a plurality of cache line entries, each entry having a fixed correspondence to a respective set of real addresses, said cache coherency mechanism further selectively determining whether to send cache line invalidation messages to the processor with which the cache is associated using state information in said processor portion of said cache line directory structure.

4. The digital data processing system of claim 3, wherein said processor portion of said cache line state directory structure contains cache line state for a plurality of caches associated with a plurality of processors, said cache coherency mechanism further selectively determining whether to send cache line invalidation messages to any of said plurality of processors using state information in said processor portion of said cache line directory structure.

5. The digital data processing system of claim 1, wherein said digital data processing system comprises a plurality of nodes, each node containing at least one processor, a respective portion of said memory, and a respective portion of said cache coherency mechanism.

6. The digital data processing system of claim 5, wherein each said respective portion of said cache coherency mechanism in each respective node maintains cache line state information for cached data having a real address in the respective portion of said memory contained in the node.

7. The digital data processing system of claim 5, wherein each said respective portion of said cache coherency mechanism in each respective node maintains cache line state information for data cached in devices contained in the node.

8. The digital data processing system of claim 1,

wherein said digital data processing system comprises a plurality of devices having respective device caches, each said device cache having a respective fixed number of slots for caching data, each slot caching a cache line of data; and

wherein said cache line state directory structure includes a plurality of portions, each portion corresponding to a respective one of said plurality of devices, each portion containing a respective fixed number of cache line entries equal to said respective fixed number of slots for caching data of the device cache to which the respective portion corresponds, each entry corresponding to a unique respective one of the respective fixed number of slots for caching data of the device cache to which the respective portion corresponds.

9. A method for maintaining cache coherency in a digital data processing system, comprising the steps of:

maintaining a cache line state directory structure, said cache line state directory structure having at least a portion corresponding to a device cache in a device of said digital data processing system, said portion containing exactly N cache line entries, wherein N is a fixed number greater than one, each entry having a fixed correspondence to a unique respective one of N slots for caching lines of data in said device cache, said device cache containing exactly N slots for caching N lines of data;

responsive to each of a plurality of data access requests, accessing said cache line state directory structure to determine whether data having a data address referenced by the request is contained in said device cache;

for each of said plurality of data access requests, determining whether to send an invalidation message to said device based on whether said step of accessing said cache line state directory determines that data having a data address referenced by the request is contained in said device cache; and

for each of said plurality of data access requests, sending an invalidation message to said device responsive to the determination made by said step of determining whether to send an invalidation message.

10. The method of claim 9, wherein said device is an I/O bridge device.

11. The method of claim 9, further comprising the steps of:

receiving a plurality of data access requests for cache lines of data from said device, each data access request from said device including data identifying a slot of said device cache in which the cache line will be stored; and

responsive to receiving each said data access request from said device, updating said cache line state directory structure by writing cache line identifying information corresponding to the data access request at the entry corresponding to the slot in which the cache line requested by the data access request will be stored.

12. The method of claim 9, wherein said step of maintaining a cache line state directory structure comprises maintaining a first portion of said cache line state directory structure corresponding to said device cache, and a second portion of said cache line state directory structure corresponding to a plurality of caches associated with a plurality of processors, said method further comprising the steps of:

responsive to each of said plurality of data access requests, accessing said cache line state directory structure to determine whether data having a data address referenced by the request is contained in any of said plurality of processors;

for each of said plurality of data access requests, determining whether to send an invalidation message to any of said plurality of processors based on whether said step of accessing said cache line state directory structure determines that data having a data address referenced by the request is contained in any of said plurality of processors; and

for each of said plurality of data access requests, sending an invalidation message to at least one of said plurality of processors responsive to the determination made by said step of determining whether to send an invalidation message to any of said plurality of processors.

13. The method of claim 9, wherein said digital data processing system comprises a plurality of nodes, each node containing at least one processor, a respective portion of said memory, and a respective portion of said cache coherency mechanism.

14. A digital data processing system, comprising:

a memory;

a plurality of processors controlling a plurality of caches for temporarily caching data from said memory;

at least one device having a device cache, said device cache having a fixed number of slots for caching data, each slot for storing a cache line; and

a cache line state directory structure having a first portion for maintaining cache line state for lines of data cached in said plurality of caches controlled by said plurality of processors, and a second portion for maintaining cache line state for lines of data cached in said device cache;

wherein said first portion of said cache line state directory structure contains a plurality of cache line entries, each entry having a fixed correspondence to a respective set of real addresses;

wherein said second portion of said cache line state directory structure contains exactly said fixed number of cache line entries, each entry having a fixed correspondence to a unique respective one of said fixed number of slots for caching data of said device cache.

15. The digital data processing system of claim 14, wherein said device is an I/O bridge device.

16. The digital data processing system of claim 14, wherein said digital data processing system comprises a plurality of nodes, each node containing at least one processor, a respective portion of said memory, and a respective portion of said cache line state directory structure.

17. The digital data processing system of claim 16, wherein each said respective portion of said cache line state directory structure each respective node contains cache line state information for cached data having a real address in the respective portion of said memory contained in the node.

18. The digital data processing system of claim 16, wherein each said respective portion of said cache line state directory structure each respective node contains cache line state information for data cached in devices contained in the node.

19. A cache coherency apparatus for a digital data processing system:

a communications interface for communicating with a plurality of devices;

a cache line state directory structure, wherein at least a portion of said cache line state directory structure corresponds to a cache having exactly N slots for caching data in a first device of said plurality of devices, wherein N is a fixed number greater than one, said at least a portion containing exactly N cache line entries, each entry having a fixed correspondence to a unique respective one of said N slots for caching data of said cache in said first device; and

cache coherence control logic which selectively generates invalidation messages responsive to events affecting the validity of cached data, said cache coherence control logic determining whether to send cache line invalidation messages to said first device using state information in said at least a portion of said cache line state directory structure corresponding to said cache in said first device.

20. The cache coherency apparatus of claim 19, wherein said first device is an I/O bridge device.

21. The cache coherency apparatus of claim 19, wherein said cache line state directory structure contains a plurality of discrete portions, including a first π portion corresponding to said cache in said first device, and a second portion corresponding to at least one cache associated with a processor, said second portion containing a plurality of cache line entries, each entry having a fixed correspondence to a respective set of real addresses, said cache coherence control logic further selectively determining whether to send cache line invalidation messages to said processor using state information in said second portion of said cache line state directory structure.

22. The cache coherency apparatus of claim 19, wherein said cache coherency apparatus is embodied in a single integrated circuit chip, said integrated circuit chip being separate from said first device.

APPENDIX OF EVIDENCE

No evidence is submitted.

APPENDIX OF RELATED PROCEEDINGS

There are no related proceedings.